Location reminder

Pavel Mašek
Faculty of Informatics and Management
University of Hradec Kralove,
Hradec Kralove, Czech Republic
pavel.masek@uhk.cz

***Abstract*— Tato práce se zabývá využitím senzorů mobilních zařízení pro zdokonalení funkcionality aplikací. Aplikace využívající těchto senzorů dokáží uživateli nabídnout lepší služby. V práci bude použit senzor pro určování polohy pomocí GPS. Tento senzor bude rozšiřovat možnosti aplikace typu reminder právě o možnost upomínání na základě polohy uživatele. Toto rozšíření bude možné parametrizovat a kombinovat s již známým upomínáním na základě času. Takto obohacená aplikace je mnohem způsobilejší plnit svůj účel a je schopna intuitivnějšího připomínání. Zároveň je aplikace vyvíjena pomocí platformy Apache Cordova a bude cílena na všechny hlavní platformy.**

Keywords-geolokace, mobilní aplikace, apache cordova, cross-platform development, sledování polohy

I. INTRODUCTION/ÚVOD

Využívání senzorů chytrých telefonů je klíčovým prvkem k zatraktivnění vyvíjené aplikace. Pokud jsou využívány správně nebo nápaditým způsobem mají podstatně větší šanci uchytit se na trhu s mobilními aplikacemi. V širším slova smyslu je možné za senzory považovat jakýkoliv zdroj informací, který předává data do nějaké řídící jednotky. Mobilní telefony doslova hýří množstvím senzorů jako je akcelerometr, který měří s jakým zrychlením se uživatel pohybuje, gyroskop, pro určení natočení telefonu a spoutou dalších.

V této práci je použit senzor pro určování polohy uživatele pomocí GPS. GPS je vojenský družicový systém pro určování polohy provozovaný Ministerstvem obrany Spojených Států amerických. S jeho použitím je možné určit přesnou polohu s přesností do deseti metrů. Mobilnní senzor pro určení polohy je hojně používán pro zkvalitnění služeb aplikací. Dobrým příkladem takových aplikací jsou aplikace pro předpověď počasí. Tyto aplikace získájí polohu uživatele a na jejím základě zobrazují informace o počasí v danné lokalitě.

Způsob jakým funguje systém GPS je pospsán v článcích (1) a (2). Články představují několik algoritmů na kterých je systém GPS založen a jakými jsou získaná data zpřesňována.

Možností jak chytře využít získanou polohu uživatele je nepřeberné množství od prostého zobrazení polohy uživatele na sociální síti až po doporučení nejlepší restaurace v okolí.

Tato práce se bude snažit vylepšit aplikaci typu reminder právě používáním dat získaných z GPS. Aplikace typu reminder může mít různá použití. Například ve článku (3) byla použita obdoba mobilní aplikace reminderu pro připomínání užívání léků pacientům s diabetem. Základní aplikace typu reminder může vypadat jako zápisník kam si uživatel poznamenává činnosti, které má udělat. Taková aplikace není ve své funkcionalitě připomíná příliš učinná. Připomíná úkoly až ve chvíli, kdy ji uživatel používá. Lepší aplikace dovolují alespoň nastavení času, kdy má být událost připomenuta. Zde se jedná o lepší nikoli však dostatečné řešení.

Použití geolokace je zásadním rozšířením k poskytovaní služeb reminderu. Geolokace umožňuje nastavit spouštění upomínek na základě polohy uživatele aplikace. Také je možná kombinace upozornění na základě polohy uživatele a času ve kterém má být událost připomenuta. Takto obohacená aplikace je chopná daleko lépe plnit svůj účel.

Aplikace bude vyvíjena pomocí platformy Apache Cordova, která umožňuje multiplatformní vývoj mobilních aplikací. Platforma využívá pro vývoj aplikací technologie HTML, CSS a Javascript. Aplikace může být tedy vyvinuta bez psaní nativního kódu konkrétní platformy.

II. Problem Definition/ Definice problému

V oblasti aplikací pro správu upomínek existuje již značná konkurence. Lze tak předpokládat podle množství různých žebříčků, které porovnávají funktionalitu jednotlivých aplikací jako jsou ve článcích (4) a (5).

Pro zhodnocení konkurenčních aplikací byl výběr přizpůsoben spíše těm, které používají upozornění na základě polohy uživatele. Předpokládané požadavky na aplikaci jsou: přesné upozorňování na nastevné události podle polohy, intuitivní uživatelské rozhraní, ohleduplné zacházení se systémovými zdroji, možnost editaci a vkládání událostí pomocí webového rozhraní a existence aplikace na všech hlavních mobilních platformách.

První z konkurenčních aplikací je aplikace pro platformu Android se jménem Geobell. Naneštěstí je aplikace vytvořena pouze pro tuto platformu. Tato aplikace vcelku přesně upozorňuje na nastavené události. Ovšem ovládání této aplikace je poměrně neintuitivní. Mezi další její nedostatky patří rovněž nepřiměřená spotřeba baterie. Dále pak nedovoluje zadávání a editaci pomocí webového prohlížeče.

Velice známou aplikací pro správu upomeník je Google Calendar. Nabízí možnost vkládání a editace upomínek pomocí webového prohlížeče a mobilní aplikace. Vlastní jednoduché a intuitivní uživatelské rozhraní. Snad jedinou jeho chybou je právě absence možnosti vkládat upomínky na základě polohy.

Pro platformu iOS je vyvinuta aplikace LocationMinder. Tato aplikace nabízí intuitivní uživatelské rozhraní i přesnou detekci polohy. Podle hodnocení uživatelů lze usoudit, že je aplikace velice oblíbná. Mezi její nevýhody patří absence webového rozhraní a skutečnost, že aplikace existuje pouze pro platformu iOS.

Poslední z výběru konkurenčních aplikací je aplikace Location reminder pro platformu Windows Phone. Aplikace má ne příliš přehledné rozhraní a celkově špatnou implementaci v případě používání gps, která má za následek menší výdrž baterie.

Výsledkem zkoumání a testování konkurenčních aplikací bylo zjištění, že žádná z aplikací nesplňuje všechny nároky, které byly požadovány. Výsledná aplikace bude inspirována částmi, které byly v konkurenčních aplikacích řešeny správně, a přinese tak kombinaci všech požadavků v jedné aplikaci. Díky použití technologie Apache Cordova bude aplikace dostupná i na všech hlavních mobilních platformách.

III. New Solution / nové řešení

Při vývoji mobilních aplikací vyvstával pro vývojáře nebo vývojová studio jeden velice nepříjemný fakt. V případě, kdy chtěli novou aplikaci cílit na všechny hlavní mobilní platformy, byly nuceni znát tři různé programovací jazyky pro tři hlavní mobilní platformy. Ve výsledku to znamená, že jedna stejná aplikace musí být napsána zvlášť pro každou platformu a neexistuje způsob jak napsaný kód znovu použít pro jinou platformu. Toto řešení sice přináší velice robustní a kvalitní řešení, ale zároveň způsobuje, že vývoj mobilních aplikací je velice drahá záležitost. O výhodách a srovnáních s nativním řešení pojednávají články (6) a (7). V součastnosti existují dva způsoby, které tento problém odstraňují.

První možnost zaštiťuje firma Xamarin s komerčním produktem stejného jména. Xamarin dovoluje psát multiplatformní mobilní aplikace v programovacím jazyku C# ze kterého je posléze generován nativní kód dané aplikace pro dannou platformu. Vývoj aplikací za pomoci produktu Xamarin je obsáhle popsán například v literatuře (8) a (9). Lituratura provede vývojáře od úplných začátku zahrující i nastavení vývojového prostředí až po implementaci aplikace typu online chat. Velkou výhodou této technologie je fakt, že využívá nativní kód dané platformy a tím je rychlejší než následující multiplatformní řešení.

Druhou možností je využití open source platformy Apache Cordova na kterém stojí například velice známý projekt pro tvorbu multiplatformních aplikací nazvaný PhoneGap. Apache Cordova využívá HTML, CSS a Javascript pro tvorbu aplikací. Využití této technologie nabízí hned několik výhod. První výhoda je stejná jako v technologii předcházející a to jednotný kód pro všechny platformy. Druhou výhodou jsou použité technologie k tvorbě aplikace. De facto by se dalo říct, že tyto technologie minimálně její základy zná každý vývojář. Pro firmu je tedy mnohem jednodušší získat zaměstnance pro tuto technologii.

Technologie Apache Cordova v podstatě funguje na principu vložení celostránkové komponenty WebView. Jedna se o komponentu zobrazující obsah internetových stránek. V této komponentě se zobrazí kód napsaný v HTML, CSS a Javascriptu. Javascriptem jsou pak na základě různých událostí volány funkce jednotlivých knihoven platformy Apache Cordova. Tyto funkce jsou převážně přidávány do globálního javascriptového objektu window.

Výsledná aplikace bude vycházet z poznatků získaných ze šetření z předešlé kapitoly. Jedním z požadavků je cílení nové aplikace na všechny hlavní platformy, kterými jsou Android, iOS a Windows Phone. Tohoto požadavku bude dosaženo na základě využití právě platformy Apache Cordova. Podrobný postup jak vyvíjet aplikace na této platformě popisuje literatura (10) a (11).

Dalším zásadním bodem je přesné upozornění na události vztahující se k poloze uživatele. Získávání polohy bude řešeno pomocí knihovny org.apache.cordova.geolocation. Způsob a možnosti používání knihovny jsou představeny v dokumentaci (12).Velice úzce souvisejícím požadavkem je požadavek na šetrné zacházení se systémovými zdroji. Tímto požadavkem je míněna nadbytečná spotřeba baterie. Nadbytečná spotřeba baterie je většinou způsobena příliš krátkým intervalem dotazování na polohu. Nastavením vhodného intervalu dotazování na polohu aplikace předejde tomuto problému. Tento interval bude dále editovatelný v nastavení aplikace pro případ, že by některému uživateli nevyhovoval.

Pro intuitivnost uživatelského rozhraní se bude jeho vývoj řídit podle článku (13), který je vztažený k vývoji multiplatformních mobilních aplikací v porovnání s vývojem

uživatelského rozhraní napsaného v nativním kódu platformy.

S intuitivnostní a vlastně celkovou snadnou ovladatelností aplikace souvisí i další požadavek. Podobně jako dovoluje konkurenční řešení Google calendar editaci a synchronizaci upomínek ve webovém prohlížeči, měla by i nově vytvořená aplikace mít toto rozšíření. Uživatel tímto rozšířením dostává další možnost jak snadněji vkládat a editovat své upomínky. Webová aplikace bude používat na serverové části Java Spring MVC a jako databázi MySQL. Aby se aplikace jevila jako běžná aplikace bude její frontendová část napsána jako single page aplikace(dále jen SPA) stějně jako Google calendar. Pro vytvoření SPA bude použit typescriptový framework pro vytváření SPA Bobril TS. Používání Bobril TS je včetně příkladů a veškerých kódů popsáno v online dokumentaci (14).

IV. Implementation / Implementace řešení

Vývoj multiplatformních aplikací pomocí frameworku apache cordova používá jako programovací jazyk javascript. Aplikace je pak vyvíjena jako webová stránka takzvaná SPA. Proto je velice vhodné používat již nějaký javascriptový framework pro vytváření SPA. Na výběr je z mnoha populárních frameworků jako jsou AngularJS nebo React.

První jmenovaný AngularJS používá standartní přístup k HTML DOMu a manipulaci s ním. React na rozdíl od AngularJS používá takzvaný virtuální DOM. V podstatě veškeré HTML elementy jsou generovány javascriptem. Velikou výhodou tohoto přístupu je mnohem větší rychlost. V Reactu je například napsaná velice populární sociální síť Facebook.

Pro implementaci aplikace nebyl zvolen ani jeden z výše jmenovaných frameworků. Jako framework byl vybrán Bobril TS. Jedná se o typescriptový framework pro vytváření SPA aplikací. Typescript je nadstavba javascriptu, která přináší typovou kontrolu, třídy, moduly rozhraní, intellisense a refactoring kódu. Framework Bobril TS je inspirován knihovnou React a tedy také používá virtuální DOM. Velkou výhodou je právě typovost, která je vítaná při vytváření rozsáhlejší aplikace. Další výhodou tohoto frameworku je jeho velikost. Zminifikovaná základní verze má velikost pod 10 kilobajtů.

Nejhlavnějším bodem celé aplikace je implementace upomínání uživatele na základě jeho lokace. Pro získání pozice uživatele je používána knihovna org.apache.cordova.geolocation. Ta zajišťuje přístup k GPS mobilního zařízení. Dovoluje implementovat metody jak pro úspěšné získání souřadnic tak i pro neúspěšné. Podrobný a kompletní popis knihovny je uveden v online dokumentaci (12).

Získávání polohy uživatele probíhá každých pět minut ve výchozím nastavení aplikace. Je tak kladen důraz na co nejpřesnější upozornění. Tento interval je možné editovat v nastavení aplikace.

Uživatel může vytvářet dva základní typy upomínek a to geolokační a časovou. Tyto dvě varianty pak může i kombinovat. V případě časové upomínky se nastavuje začátek události a konec události. V případě geolokační upomínky je nastavení o něco složitější. Vložit geolokační upomínku je možné pomocí dvou způsobů. Prvním z nich je napsání přesné adresy místa připomenutí. Druhou možností je vybrání místa pomocí mapy. Pro zobrazení mapy se využívá knihovna Google Maps Javascript API v3.

Dalším parametrem je nastavení upozornění při příchodu do daného místa nebo při odchodu z něj. Posledním parametrem nastavení je zvolení radiusu pro upozornění.

Kombinací geolokační a časové upomínky je možné dosáhnout velice přesného upozornění. Například při odjezdu na dovolenou „Připomeň 20.6.2015 v čase 8:00 až 10:00 ve vzdálenosti 500 metrů od domova – Vzal jste si pasy?“.

Výpočet vzdálenosti mezi dvěma GPS souřadnicemi je počítám pomocí haversinovi formule. Pro tento výpočet slouží malá javascriptová knihovna haversine.js. Používání této knihovny je popsáno v online dokumentaci (15).

Aplikace si stále udržuje seznam událostí. Geolokační události jsou kontrolovány, zdali nenastali, hned po získání nových souřadnic. Tedy podle výchozího nastavení každých pět minut. Zároveň se uchovává právě jedna předchozí pozice uživatele, aby bylo možné zjistit, zda opustil místo přípomenutí. Pro každou geolokační událost se vypočítá vzdálenost od polohy uživatele. Tato vzdálenost je pak porovnávána s nastaveným radiusem události. V případě události příchodu do místa musí být pro spuštění upomínky vzdálenost menší než radius. V případě odchodu z místa musí být aktuální vzdálenost větší než radius a zároveň předchozí vzdálenost musí být menší než radius danné události.

Zadávání událostí je umožněno i přes webové rozhraní. Je proto nutná synchronizace událostí v mobilním zařízení s databází webové aplikace. Automatická synchronizace je ve výchozím nastavení nastavena na patnáctiminutový interval. Tato automatická synchronizace probíhá jenom ve chvíli, kdy mobilní zařízení není používáno. Dále se synchronizace vyvolá vždy při spuštění nebo znovu otevření aplikace.

Samotná synchronizace probíhá tak, že aplikace pošle serveru čas poslední synchronizace, kterou má uloženou v parametru uživatele. Každý uživatel má parametr poslední modifikace jak lokálně na mobilním zařízení tak na serveru.

Server porovná čas poslední aktualizace zaslané mobilním zařízením s časem uloženým v databází. Ve chvili kdy se tyto dva časy liší server vygeneruje pole všech aktivních událostí a ty zašle zpátky na mobilní zařízení spolu s novým časem poslední aktualizace. Mobilní zařízení zařízení si aktualizuje čas poslední modifikace ve svém lokálním úložišti a poté začne porovnávat události. Každá událost má unikátní ID vygenerované databází serverové části, které slouží jako jednoznačný identifikátor. Pomocí tohoto ID jsou události porovnávány. Události, které mají stejné ID a liší se jejich hodnoty jsou aktualizování v mobilním zařízení podle událostí obdržených ze serveru. Ty události, které jsou uloženy pouze v mobilním zařízení a nejsou v poli událostí vygenerovaných serverem jsou smazány. A události, které nejsou uloženy v mobilním zařízení a jsou v poli vygenerovaném serverem jsou přidány jako nové události.

Persistentní ukládání událostí v mobilním zařízení je řešeno pomocí ukládání json souboru. Json soubor byl zvolen z důvodu, že reprezentuje přirozenou stavbu objektu jazyka javascript. Pro přístup k úložišti mobilního zařízení je použit plugin org.apache.cordova.file. Pomocí něho lze snadno číst a zapisovat soubory do úložiště mobilního zařízení. Kompletní dokumentace s příklady použití je dostupná online na (16).

V. TESTING OF DEVELOPED APPLICATION / TESTOVÁNÍ VYVINUTÉ APLIKACE - ŘEŠENÍ

Testování aplikace probíhalo v několika fázích. Z počátku vývoje bylo testováno uživatelské rozhraní, později se přidalo testování ukládání dat spolu s testováním získávání polohy. Pro vhodné, komplexnější a často se opakující se problémy bylo využito unit testování a to konkrétně testovací framework jasmine pro testování javasriptu.

Nakonec proběhlo závěrečné krátké UX testování spolu s nasazením testovací verze aplikace na tři různé mobilní zařízení.

V první fázi vývoje bylo testováno uživatelské rozhraní. Testovány byly všechny prvky, které vyvolávaly alespoň nějakou interakci. Především tlačítka, otevírání levého menu, touch události a swipe gesta.

Na základě výsledku testování první fáze bylo upuštěno od animování otevírání levého menu pomocí CSS3 vlastnosti transation. Animace nebyla plynulá a nebudila dojem fungující aplikace.

U konkurenčních aplikací používajících nativní kód dané platformy nebyl žádný postřehnutelný problém s plynulostí animace.

Další fází bylo testování persistentního ukládání dat a jejich načítání z úložiště mobilního zařízení. Výsledkem testování měl být stejný seznam událostí, jako byl vytvořen před začátkem testování.

Testovanými případy bylo uložení při standartním vypnutí aplikace, vypnutí aplikace pomocí správce úloh a při vypnutí telefonu.

Ve všech případech testování persistentního ukládání byl seznam událostí po testování stejný jako seznam událostí před testováním.

Po úspěšném testu persistentního ukládání byl na řadě test synchronizace událostí. Byly vytvořeny testovací sady pro testování veškerých možných případů, které by mohli nastat. Ke každé testovací sadě byl vytvořen správný výsledek, kterého mělo být dosaženo.

Mezi testované případy patřily situace: žádná událost není uložena v mobilním zařízení(může jít o počáteční stav mobilní aplikace), žádná událost není uložena na serveru (případ, kdy aplikace začala být používána), zařízení má události a server vygeneruje prázdné pole(zde mohou nastat dva stavy pokud je čas aktualizace rozdílný měly by být všechny události v mobilním zařízení smazány, pokud je čas aktualizace stejný nejsou žádné události k aktualizaci, přidání nebo odebrání), server vygeneruje více událostí než jich je uložených v mobilním zařízení(zde se může jednat buďto o pouhé přidání nových událostí přes webové rozhraní a nebo o přidání nových a odebrání menší počtu starých událostí než bylo přidáno) a poslední případem je vygenerování menšího počtu událostí než jich je uložených v mobilním zařízení (zde se opět může jednat buďto o prosté odebírání událostí a nebo o přidání menšího počtu nových událostí).

Pro tuto fázi testování bylo použito unit testování za pomoci testovacího frameworku jasmine. Dokumentace k testovacímu frameworku jasmine včetně ukázek kódu je dostupná online na (17).

Předposlední faze testování byla spojena s testováním správnosti upozornění využívající polohu uživatele. Pro toto testování byl použit Ripple emulator, který dovoluje testovat aplikace typu apache cordova přímo v desktopovém prohlížeči. Ripple emulator je možné nainstalovat jako balíček node.js nebo je možné ho přidat jako plugin do prohlížeče google chrome.

Ripple emulator dovoluje simulovat mnoho stavů do kterých se může reálné mobilní zařízení dostat včetně simulace pohybu telefonu nebo přístupu k síti. Dovoluje také simulovat polohu uživatele což velice usnadnilo a výrazně urychlilo vývoj i testování aplikace.

Při testování byla opět vytvořena testovací sada dat se správnými výsledky. Po výpočtu upozornění se musely výsledky z testovací sady shodovat s výsledky výpočtu.

Správnost upozorňování byla testována rovněž uživateli na třech různých mobilních zařízeních. Testovací zařízení byly mobilní telefony se systémem Android od Samsungu a to Galaxy S2(Android 4.2), Galaxy S3(Android 4.4) a Galaxy S4(Android 4.4).

Pro sběr dat ze zpětné vazby bylo upraveno vyskakovací okno upozorňující na událost. Do okna byla přidána dvě tlačítka pro kladné a záporné hodnocení.

Výsledky testu potvrdili téměř bezchybné zobrazení vyskakovacího okna s upozorněním ve chvíli, kdy uživatel dosáhl určité oblasti. Zároveň se ale vyskytly i chyby v upozornění. Ve chvíli, kdy uživatel zadal velice malý radius(konkrétně 100 metrů a méně), mohlo se stát, že stihl projít oblastí dříve než si aplikace znovu vyžádala polohu. Z tohoto důvodu bude muset být upraven čas pro vyžádání polohy na co nejmenší interval, ale s ohledem na výdrž baterie.

Poslední fází testování bylo uživatelské UX testování. Uživatelského testování se zúčastnili dva uživatelé. Byla jim představena aplikace a vysvětleno co aplikace ovládá. Poté jim byla aplikace nainstalována na jejich mobilní zařízení. Bez jakéhokoliv návodu jim byly zadány úkoly. Oba uživatelé dostali stejné úkoly. Nejprve měli vložit dvě čistě časové události, poté čtyři události vztažené k určité poloze a nakonec dvě události kombinované (geolokační a zároveň časová).

S vytvářením časové události neměli uživatelé nejmenší problémy. Oba tento úkol splnili správně a ve velice krátkém čase. V případě vytváření geolokační události se vyskytl problém při využívání zadání místa pomocí mapy. V tomto kroku není uživatelský interface poskládán zcela jasně, a proto by bylo vhodné logiku zadávání pomocí mapy upravit. Při vytváření kombinované události žádné nové problémy nevyvstali.

Z uživatelského testování dále vyplynula změna výchozího zobrazení aplikace. Uživatelé podotýkali, že se jedná o geolokační aplikaci, ale při spuštění aplikace se nejdříve zobrazí kalendář s časovými upozorněními. Z tohoto důvodu bylo do nastavení aplikace přidáno nastavení výchozího zobrazení.

VI. Conclusions / Závěry

Aplikace byla vyvíjena jako aplikace typu apache cordova. Díky tomuto přístupu bylo možné vytvořit aplikaci pro všechny tři hlavní platformy s jednotným kódem. Tento přístup by mohl být vhodným způsobem jak snížit náklady na vývoj mobilních aplikací a zároveň čas na vývoj.

Aplikace by mohla doplnit trh s remindery. Na rozdíl od konkurenčních řešení obsahuje jak webové rozhraní, tak geolokaci a je vytvořená pro všechny tři hlavní mobilní platformy, které jsou iOS, Android a Windows phone.

Aplikace do jisté míry splňuje požadavky vytyčené na začátku vývoje. Nicméně v současném stavu by nebyla schopná konkurovat již vytvořeným řešením. Po grafické stránce nebudí nejlepší první dojem. Dále by bylo vhodné zapracovat na lepším upozornění v případě výběru malého radiusu. V neposlední řadě upravit uživatelské rozhraní více na míru uživatelům pomocí dalšího UX testování.

VII. References / Reference

1. **Kegen, Yu a Eryk, Dutkiewicz.** Correction to "Geometry and Motion-Based Positioning Algorithms for Mobile Tracking in NLOS Environments. 2012, stránky 704-704.

2. **Kegen, Yu, a další.** Real-time mobility tracking algorithms for cellular networks based on Kalman filtering. *IEEE Transactions on Mobile Computing.* 2005, stránky 195-208.

3. **Samir, Patel, a další.** Mobilizing your medications: an automated medication reminder application for mobile phones and hypertension medication adherence in a high-risk urban population. *Journal of diabetes science and technology.* 2013, Sv. 7, 3.

4. **Kazmucha, Allyson.** http://www.imore.com/. *iMore.* [Online] 14. 7 2014. [Citace: 18. 12 2014.] http://www.imore.com/best-task-and-reminder-apps-mac-goodtask-clear-due-and-more.

5. **http://www.technobezz.com/.** ***TECHNOBezz.*** **[Online] 19. 6 2014. [Citace: 18. 12 2014.] http://www.technobezz.com/6-top-best-android-apps-reminders/.**

6. ***Evaluating Cross-Platform Development Approaches for Mobile Applications.*** **Heitkötter, Henning, Hanschke, Sebastian a Majchrzak, Tim A. Sv. 120.**

7. Charland, Andre a Leroux, Brian. Mobile application development. ***Communications of the ACM.*** **2011, Sv. 54, 5.**

8. ***Xamarin crossplatform application development.*** **S.l. : Packt Publishing Limited, 2014.**

9. Reynolds, Mark. ***Xamarin Mobile Application Development for Android.*** **Birmingham : Packt Publishing, 2014.**

10. Shotts, Kerri. ***PhoneGap Social App Development.*** **Birmingham : Packt Pub, 2013.**

11. —. ***PhoneGap 2.x Mobile Application Development.*** **Birmingham : Packt Pub., 2013.**

12. Soref, Josh. www.github.com/apache/cordova-plugin-geolocation. ***www.github.com.*** **[Online] 6. 11 2014. [Citace: 18. 12 2014.] https://github.com/apache/cordova-plugin-geolocation/blob/master/doc/index.md.**

13. ***Mobile application development, creating exciting apps for mobile devices using PhoneGap.*** **Shotts, Kerri, Charland, Andre a Leroux, Brian. Birmingham : Packt Pub., 2013.**

14. https://github.com/Bobris/Bobril. ***https://github.com/Bobris/Bobril.*** **[Online]**

15. GitHub. ***www.github.com.*** **[Online] https://github.com/niix/haversine.**

16. apache/cordova-plugin-file. ***www.github.com.*** **[Online] https://github.com/apache/cordova-plugin-file/blob/master/doc/index.md.**

17. jasmine/jasmine. ***www.github.com.*** **[Online] https://github.com/jasmine/jasmine.**

18. ***Portability of mobile applications using phonegap.*** **Manoharan, R., a další. 2012, Sv. 23-23.**

19. Harrington, Chris. Code Mentor. ***www.codementor.io.*** **[Online] https://www.codementor.io/reactjs/tutorial/reactjs-vs-angular-js-performance-comparison-knockout.**